Iranian Journal of Fuzzy Systems

Volume 17, Number 4, (2020), pp. 7-24

University of Sistan and Baluchestan

Fuzzy particle swarm optimization with nearest-better neighborhood for multimodal optimization

M. B. Dowlatshahi[1], V. Derhami[2] and H. Nezamabadi-pour[3]

[1] *Department of Computer Engineering, Faculty of Engineering, Lorestan University, Khoramabad, Iran.*
[2] *Department of Computer Engineering, Faculty of Engineering, Yazd University, Yazd, Iran.*
[3] *Department of Electrical Engineering, Shahid Bahonar University of Kerman, Kerman, Iran.*

dowlatshahi.mb@lu.ac.ir, vderhami@yazd.ac.ir, nezam@uk.ac.ir

Abstract

In the last decades, many efforts have been made to solve multimodal optimization problems using Particle Swarm Optimization (PSO). To produce good results, these PSO algorithms need to specify some niching parameters to define the local neighborhood. In this paper, our motivation is to propose the novel neighborhood structures that remove undesirable niching parameters without sacrificing performance. Hence, this paper has two main contributions. First, two novel parameter-free neighborhood structures named Topological Nearest-Better (TNB) neighborhood and Distance-based Nearest-Better (DNB) neighborhood are proposed in the topological space and decision space, respectively. Second, two proposed neighborhoods are combined with Fuzzy PSO (FPSO) and two novel niching algorithms, called TNB-FPSO and DNB-FPSO, are proposed for solving multimodal optimization problems. It should be noted that we use a zero-order fuzzy system to balance between exploration and exploitation in the proposed algorithms. To evaluate the performance of proposed algorithms, we performed a detailed empirical evaluation on the several standard multimodal benchmark functions. Our results show that DNB-FPSO statistically outperforms the other compared multimodal optimization algorithms.

Keywords: Particle swarm optimization, topological nearest-better neighborhood, distance-based nearest-better neighborhood, multimodal optimization, fuzzy balancer.

1 Introduction

Particle Swarm Optimization (PSO) is one of the most widely used swarm intelligence algorithms for solving optimization problems [30]. The original version of PSO is merely ideal for solving unimodal optimization problems, hence it is not recommended to use the original PSO to solve a multimodal optimization problem because this algorithm tends to converge to the best solution of the problem [37, 38]. The main reason is that the position of the best global particle is used to update the velocity of all particles in each iteration. In this case, all particles are simultaneously absorbed into the position of the global best particle, and therefore the swarm will gradually converges to the global best particle. In the last decades, great efforts have been made to solve multimodal optimization problems using PSO [37]. To prevent the convergence of the swarm to the best solution, all of these PSO algorithms are equipped with "niching" techniques to preserve the swarm diversity. A common limitation has been added to all of these PSO niching algorithms by which each particle connects only to several other particles in "its local neighborhood"and is attracted only by the particles of this local neighborhood. It is worth mentioning that the neighborhood can be defined either in the topological space or decision space.

To produce good results, most niching PSO algorithms need to specify some niching parameters to define the local neighborhood. The most representative niching parameter is niche radius, which should be determined to demonstrate how far apart the optima are from each other. Because the performance of these algorithms depends on determining

Corresponding Author: M. B. Dowlatshahi
Received: April 2019; Revised: September 2019; Accepted: November 2019.

the exact value of these niching parameters, they are not easily usable in practice. For example, using a small niche radius causes a particle to move based on the position of its few neighbors, and consequently those particles which are best in a local neighborhood eventually form many niches. It should be noted that some of these niches may not be real niches. Instead, the use of a large niche radius allows a particle to move based on the position of more particles and consequently there is a risk that we may lose some real niches [37]. Therefore, our motivation is to propose new neighborhood structures for PSO that remove undesirable niching parameters without sacrificing performance.

The main contributions of this paper can be summarized as follows:

- Two novel local neighborhood named Topological Nearest-Better (TNB) neighborhood and Distance-based Nearest-Better (DNB) neighborhood are proposed in the topological space and decision space, respectively. In both proposed local neighborhoods, each particle i is connected to a particle j such that: 1) the objective function value of j is better than that of i, and 2) the particle j is at least as close as to particle i than any other particle k that its quality is better than particle i.
- Two proposed local neighborhoods are used in Fuzzy PSO (FPSO) and two effective parameter-less niching algorithms, called TNB-FPSO and DNB-FPSO, are proposed for solving multimodal optimization problems. The TNB-FPSO and DNB-FPSO do not require specification of any niching parameters such as how far apart between two optima or the number of optima.

The structure of this paper is organized as follows. In Section 2, the original versions of PSO and the important multimodal optimization algorithms that were previously introduced are reviewed. In Section 3, two proposed TNB-FPSO and DNB-FPSO are introduced. Section 4 contains the experimental setups and experimental results of the paper. Finally, conclusion and future works are given in Section 5.

2 Scientific background and related works

In this section, we first introduce the basic concepts of PSO. Next, we explain the different types of PSO to solve multimodal optimization problems. Finally, we review the other well-known multimodal optimization algorithms.

2.1 Particle Swarm Optimization (PSO)

An optimization problem involves maximizing or minimizing an objective function by systematically selecting input values from within an allowed set and calculating the value of the function [7, 28, 55]. The generalization of optimization theory and techniques to other formulations constitutes a large area of applied mathematics [53, 54]. More generally, optimization includes finding "best available" values of some objective function given a defined domain (or input), including a variety of different types of objective functions and different types of domains[56, 62]. Swarm intelligence algorithms are stochastic optimization methods that mimic the natural biological evolution and/or the social behavior of species [12, 15, 16]. The class of swarm intelligence algorithms includes, but is not restricted to, Particle Swarm Optimization (PSO) [19, 17, 18, 21, 40], Competitive Swarm Optimization (CSO) [13], Gravitational Search Algorithm (GSA) [14, 20, 49], and Ant Colony Optimization (ACO) [11].

In 1995, Kennedy and Eberhart [30] proposed the PSO algorithm to solve continuous optimization problems. Let we show each particle as a triple $Particle_i = (\vec{X}_i(t), \vec{V}_i(t), \overrightarrow{Pbest_i}(t))$ in an n-dimensional search space where $\vec{X}_i(t)$ and $\vec{V}_i(t)$ are the position and velocity vectors of the ith particle, respectively, and $\overrightarrow{Pbest_i}(t)$ is the vector of personal best position found by the particle as follows [30]:

$$\vec{X}_i(t) = (x_i^1(t), x_i^2(t), \ldots, x_i^d(t), \ldots, x_i^n(t)) \quad for\ i = 1, 2, \ldots, N. \tag{1}$$

$$\vec{V}_i(t) = (v_i^1(t), v_i^2(t), \ldots, v_i^d(t), \ldots, v_i^n(t)) \quad for\ i = 1, 2, \ldots, N. \tag{2}$$

$$\overrightarrow{Pbest_i}(t) = (pbest_i^1(t), pbest_i^2(t), \ldots, pbest_i^d(t), \ldots, pbest_i^n(t)) \quad for\ i = 1, 2, \ldots, N. \tag{3}$$

where N is the number of particles (i.e. swarm size). Also, suppose the best position of the swarm is shown by vector $\overrightarrow{Gbest}(t)$ as follows:

$$\overrightarrow{Gbest}(t) = (gbest^1(t), gbest^2(t), \ldots, gbest^d(t), \ldots, gbest^n(t)) \tag{4}$$

The next velocity of the ith particle is calculated as follows [30]:

$$v_i^d(t+1) = w(t) \times v_i^d(t) + c_1 \times r_1 \times (pbest_i^d(t) - x_i^d(t)) + c_2 \times r_2 \times (gbest^d(t) - x_i^d(t)), \tag{5}$$

where $w(t)$ is inertia weight, $c_1 > 0$ and $c_2 > 0$ are acceleration coefficients, r_1 and r_2 are random numbers in the interval [0,1], and $v_i^d(t+1)$ and $v_i^d(t)$ are the next and current velocity of the ith particle, respectively. It should be noted that $w(t)$ manage the balance between exploration and exploitation, so that a larger value of this parameter encourages exploration, while a smaller value of it provides exploitation. To do a good balance between exploration and exploitation, we propose a fuzzy inference system to adjust the value of $w(t)$ in Section 3.3.

To update the next position of the ith particle, the current position of this particle, i.e. $x_i^d(t)$, and its next velocity, $v_i^d(t+1)$, is used as follows [30]:

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1). \tag{6}$$

2.2 Related works for multimodal optimization

As its name implies, a multimodal optimization problem has several global optima, or just several global optima and multiple local optima [37, 38]. It can be said that most of the optimization problems in the real world are multimodal, such as metabolic network modelling [32], femtosecond laser pulse shaping problem [60], job shop scheduling problem [42, 43], resource constrained multi-project scheduling problems [44], automatic point determination [10], seismological inverse problem [31], Monte Carlo nonlinear filtering [1], image segmentation [6], clustering [9, 52], feature selection [27], real-time tracking of body motion [69], competitive facilities location and design [50], solving systems of equations [61], protein structure prediction [65], induction motor design for electric vehicle [8], learning multimodal parameters of neural network ensembles [22], and so on. Recently, many efforts have been done to solve multimodal optimization problems, which can be divided into two classes: depending on niching parameters and parameters independent. In the former class, users must define the niching parameters to determine the size of each niche. The most important weakness of these algorithms is their dependence on the niche parameters, because the definition of parameters depends heavily on the multimodal optimization problem being solved.

2.2.1 Parameter-dependent niching PSO

Brits et al. [5] proposed NichePSO by using sub-swarm in which several sub-swarms are created from an initial swarm by monitoring the particles'fitness. Next, the performance of NichePSO is increased by Ozcan et al. [39] using fanaticism and climbing techniques. Li proposed a Species-based PSO (SPSO) [33] in which each species and its corresponding dominant particles form a separate sub-swarm. SPSO requires to set a niching radius r to define the size of a niche. At each generation, all particles are sorted in descending order by their fitness value. Then, by calculating the Euclidean distance between the two arbitrary particles, the algorithm determines whether or not the two particles are within the same niche. For example, two particles are in the same niche if the Euclidean distance between them is smaller than r.

After dividing the niches, the algorithm updates each particles *gbest* in each niche by the best particles *pbest* in this niche. The swarm probably traps into local optima or the algorithm may miss some global optima if r is too small. Iwamatsu and Masao [24] proposed a modified PSO called Multi-species PSO (MSPSO) which has the same idea as SPSO expect that MSPSO sorts all the particles in descending order by their personal best fitness values at each generation. Bird and Li proposed a version of SPSO called Enhanced SPSO (ESPSO) [3] which enhances SPSO by increasing the robustness of the niching parameter to the point that the algorithm is still effective even if it is not used at all.

Passaro and Starita [41] developed a novel niching PSO, called k-means-based PSO (kPSO), which identify niches by clustering particles using the standard k-means clustering and Bayesian information criterion. Seo et al. [58] developed a novel niching PSO to solve multimodal function optimization problems, called Multi-Grouped PSO (MGPSO), which gives a territory to every group to avoid overlapping of discovered solutions. In MGPSO, if the radius became too small before sufficient convergence level, some groups cannot find their own solutions and wandered around other groups'solutions. To overcome this defect, they proposed another algorithm based on MGPSO called Auto-Tuning MGPSO (AT-MGPSO) [59]. In AT-MGPSO, a competition mechanism is invited and all the groups have a different radius.

Qu et al. [48] developed a distance-based locally informed PSO (LIPS) which uses several local best positions to guide the search direction of each particle instead of the global best position. LIPS can operate as a stable niching algorithm using the information provided by particles'neighborhoods assessed by Euclidean distance. Qu et al. [47] compare LIPS with FER-PSO, RPSO, SPSO, and several state-of-the-art evolutionary multimodal optimizers on several commonly used multimodal benchmark functions. The experimental results suggest that LIPS can provide statistically superior and more consistent performance over the niching algorithms on the test functions, without incurring any severe computational burdens.

2.2.2 Parameter-less niching PSO

Li [34] proposed a multimodal PSO based on Fitness Euclidean-distance Ratio (FER-PSO). In this algorithm, particles select the individual's *gbest* whose fitness value has changed the most greatly in unit distance according to FER as the attractor when particles update velocities. FER-PSO needs to compute $O(N^2)$ (N is the swarm size) FER in each iteration. Although FER-PSO does not require the specification of niching parameters, it introduces a new parameter α, which needs to be determined from the search range of each variable.

Li [35] proposed a new niching PSO based on Ring topology (RPSO). The algorithm does not depend on any niching parameter. He has demonstrated that the PSO algorithms with ring topology can induce more stable niching behavior. The algorithm uses a PSO with ring topology to guide the particles in which each particle interacts only with its immediate neighbors. Different niches are formed by using the ring topology and subsequently the optimization of multiple peaks are realized. One major disadvantage of RPSO is that the ring topology links members from different niches also. If the neighbors are not from the same niche targeting a single peak, it is difficult for the niching algorithm to converge and locate the peaks effectively. In the other words, if two particles are from different niches, they may oscillate between two peaks. Note that the oscillation between two niches is unfavorable for exploration and exploitation. However, Li has demonstrated that RPSO can form stable niches across different local neighborhoods, eventually locating multiple global/local optima on a set of selected problems.

2.2.3 Other recent parameter-less niching algorithms

Recently, to eliminate the dependence of the niching algorithms on niching parameters, some researchers have proposed the idea of converting multimodal optimization problem into a multi-objective optimization problem and then solving the obtained multi-objective optimization problem by a multi-objective evolutionary algorithm. Wang et al. [63] proposed a method to convert a multimodal optimization problem into a bi-objective optimization problem with two conflicting objectives. They showed that after the above conversion, all optimal solutions of the multimodal optimization problem are placed on the Pareto front of the bi-objective optimization problem. Therefore, they have used the well-known NSGA-II to solve the bi-objective optimization problem to find the optimal solutions of the multimodal optimization problem.

Yu et al. [66] proposed a novel method to transform multimodal optimization problem into a tri-objective optimization problem with three conflicting objectives as follows: 1) the objective function of multimodal optimization problem, 2) the Euclidean distance between each solution and a set of reference points, and 3) the shared fitness of each solution based on niche count. They mathematically proved that after the above conversion, all optimal solutions of the multimodal optimization problem are placed on the Pareto front of the tri-objective optimization problem. Therefore, they have proposed a tri-objective differential evolution algorithm to solve the bi-objective optimization problem to find the optimal solutions of the multimodal optimization problem.

3 Proposed algorithms

In this section, two proposed PSO-based algorithms for solving multimodal optimization problems are proposed. First, Topologically Nearest-Better Fuzzy PSO (TNB-FPSO) is described in subsection 3.1 Then, Distance-based Nearest-Better Fuzzy PSO (DNB-FPSO) is described in subsection 3.2

3.1 TNB-FPSO: Topological Nearest-Better Fuzzy PSO

In PSO, velocity guides the search direction of the particles. In the original form of PSO, the velocity vector for a particle i is updated according to four other vectors that are the current position of the particle, the personal best position of the particle, the global best position found over the whole swarm, and the previous velocity of the particle. In contrast to original PSO, a new form of the velocity updating rule is introduced in the proposed TNB-FPSO in which "the topologically nearest-better position" rather than "the global best position" is used to guide the search direction of the particles. In the other words, in TNB-FPSO, the velocity vector for particle i is updated according to the current position of the particle, the personal best position of the particle, and the topologically nearest-better position found over the neighborhood of the personal best position of the particle.

In proposed TNB neighborhood, each particle i is connected to a particle j so that: 1) the objective function value of particle j is better than particle i, and 2) the particle j is topologically at least as close as to particle i as any other particle k that its quality is better than particle i. An example of TNB neighborhood for a minimization problem is illustrated in Fig. 1.

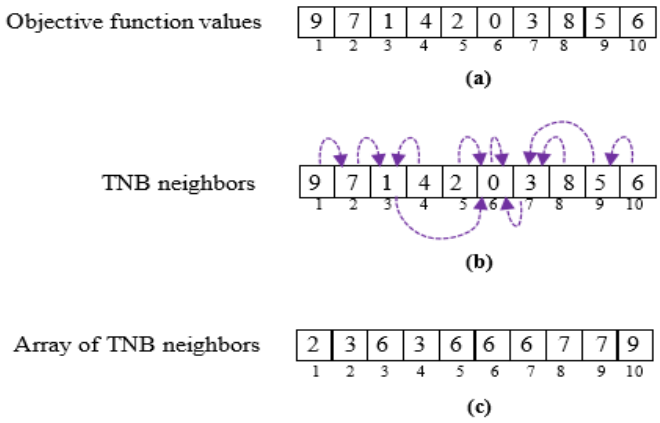

Fig 1. An example of TNB neighborhood for a minimization problem. (a) Objective function value of each particle is inside the cell corresponding to that particle. (b) Above and below the array of objective function values are arrows showing TNB neighborhood. Each particle is connected to its TNB neighbor. (c) The number within each cell of the array is TNB neighbor of this cell. For example, the number 2 within cell 1 means that particle 2 is the TNB neighbor of particle 1.

The velocity update rule of TNB-FPSO uses the formula given below while the position update rule is the same as the position update rule of original PSO (i.e. Eq. (6)):

$$v_i^d(t+1) = w(t) \times v_i^d(t) + c_1 \times r_1 \times (pbest_i^d(t) - x_i^d(t)) + c_2 \times r_2 \times (TNB_i^d(t) - x_i^d(t)), \tag{7}$$

where $\overrightarrow{TNB}_i(t) = (TNB_i^1(t), \ldots, TNB_i^d(t), \ldots, TNB_i^n(t)$ is the TNB personal best position over the neighborhood of the personal best of particle i. Algorithm (1) can find all TNB neighbors, i.e. for each particle i, find its TNB neighbor. As can be seen, this algorithm consists of three main phases.

In phases 1 and 2, the Left TNB (LTNB) neighbors and the Right TNB (RTNB) neighbors are found by the stack data structure, respectively. In phase 3, the TNB neighbors are found by scanning LTNB and RTNB.

To determine how efficiently Algorithm (1) find all TNB neighbors, we need to analyze the time complexity of the algorithm. Since three main phases of the algorithm are independent, so the time complexity of each phase should be analyzed separately, and ultimately, the complexity of the algorithm is equal to the sum of the complexity of all phases. In the following, the complexity of the time of each phase is observed.

The basic operation of phase 1 of Algorithm (1) is "Pop(S)" instruction. Despite a nested loop in phase 1, its time complexity is linear in term of swarm size, i.e. $O(N)$, because every step of the inner loop pops an item that had been added in some previous step of the outer loop. In the other words, in the worst case the maximum number of executing the instruction "Pop(S)" is of order $O(N)$, because in the worst case we push at most N item into the stack. This analysis is true for phase 2 of the Algorithm (1).

The basic operation of phase 3 is "If-else" instruction which its time complexity is linear in term of swarm size, i.e. $O(N)$. Therefore, the growth order of Algorithm (1) is $O(N)$.

The pseudo-code of the TNB-FPSO is described in Algorithm (2).

3.2 DNB-FPSO: Distance-based Nearest-Better Fuzzy PSO

In contrast to TNB neighborhood, the DNB neighborhood is defined in the decision space. In proposed DNB neighborhood, each particle i is connected to a particle j so that: 1) the objective function value of particle j is better than particle i, and 2) in decision space, the particle j is at least as close as to particle i than any other particle k that its quality is better than particle i. An example of DNB neighborhood is illustrated in Fig. 2. As can be seen, in contrast to TNB neighborhood that some particles must have topological neighbors from different niches, in DNB neighborhood the neighbor of a particle is likely to form from the similar area or the same niche. Note that moving towards the DNB neighbor can satisfy two important criteria in multimodal optimization, i.e. convergence and diversity. We can gain the convergence because the DNB neighbor encourages a particle to move toward a better solution; and we can gain

Algorithm (1): Outline of finding all TNB neighbors for minimization problems.

Inputs: The position of all particles, i.e. $Pos(t) = \overrightarrow{Pbest}_1(t), \overrightarrow{Pbest}_2(t), \ldots, \overrightarrow{Pbest}_N(t)$.
//Phase 1: Finding the Left TNB (LTNB) neighbors.
$S \neq \emptyset$, $//S$ is a stack
for *i=1 to N* **do**
  while $S \neq \emptyset$ **do**
    $j = Top(S)$;
    if $f(\overrightarrow{Pbest}_i(t)) \leqslant f(\overrightarrow{Pbest}_j(t))$
    then
      $Pop(S)$
    else
      Break the while loop;
    end
  end
  if $S = \emptyset$ **then**
    $LTNB_i = -\infty$
  else
    $LTNB_i = Top(S)$;
  end
end
// Phase 2: Finding the Right TNB (RTNB) neighbors.
for *i=N to 1* **do**
  while $S \neq \emptyset$ **do**
    $j = Top(S)$;
    if $f(\overrightarrow{Pbest}_i(t)) \leqslant f(\overrightarrow{Pbest}_j(t))$
    then
      $Pop(S)$
    else
      Break the while loop;
    end
  end
  if $S = \emptyset$ **then**
    $RTNB_i = +\infty$
  else
    $RTNB_i = Top(S)$;
  end
  Push i onto S;
end
// Phase 3: Finding the TNB neighbors by LTNB and RTNB.
for *i=1 to N* **do**
  if $LTBN_i == -\infty$ ***AND*** $RTBN_i == +\infty$ **then**
    $TNB_i = i$;
  else
    if $i - LTNB_i < RTNB_i - i$ **then**
      $TNB_i = LTNB_i$;
    else
      if $i - LTNB_i > RTNB_i - i$ **then**
        $TNB_i = RTNB_i$
      else
        if $i - LTNB_i == RTNB_i - i$ ***and*** $f(\overrightarrow{Pbest}_{LTNB_i}(t)) \leqslant f(\overrightarrow{Pbest}_{RTNB_i}(t))$ **then**
          $TNB_i = LTNB_i$;
        else
          $TNB_I = RTNB_i$
        end
      end
    end
  end
end
Output: $\overrightarrow{TNB}(T) = (TNB_1, TNB_2, \ldots, TNB_N)$.

Algorithm (2): Outline of TNB-FPSO for minimization problems.

Initialize w, N, and stopping criterion;
$t = 0$;
for $i = 1$ *to* N **do**
  Randomly generate the initial solution $\vec{X}_i(t)$;
  $\overrightarrow{Pbest}_i(t) = \vec{X}_i(t)$;
  Randomly generate the initial velocity $\vec{V}_i(t)$;
end
while *stopping criterion is not satisfied* **do**
  Calculate $\overrightarrow{TNB}(t) = (TNB_1, TNB_2, \ldots, TNB_N)$ by Algorithm (1);
  for $i = 1$ *to* N **do**
    for $d = 1$ *to* N **do**
      Calculate the next velocity $v_i^d(t+1)$ by Eq. (7);
      Update the next position $x_i^d(t+1)$ by Eq. (6);
    end
    if $f(\vec{X}_i(t+1)) < f(\overrightarrow{Pbest}_i(t))$ **then**
      $\overrightarrow{Pbest}_i(t+1) = \vec{X}_i(t+1)$;
    else
      $\overrightarrow{Pbest}_i(t+1) = \overrightarrow{Pbest}_i(t)$;
    end
  end
  $t = t + 1$;
  Update $w(t)$ by fuzzy balancer;
end
Output: $\vec{X}_i(t), i = 1, \ldots, N$.

the diversity because the DNB neighbor encourages a particle to move toward a near solution as possible. Note that the probability for particle and its nearest-better neighbor to belong to different niches is low.

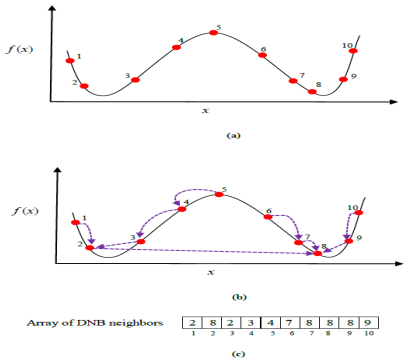

Fig 2. An example of DNB neighborhood for a minimization problem. (a) Objective function value of each particle is inside the cell corresponding to that particle. (b) Arrows are showing the DNB neighborhood. Each particle is connected to its DNB neighbor. (c) The number within each cell of the array is the DNB neighbor index of this cell. For example, the number 2 within cell 1 means that particle 2 is the DNB neighbor of particle 1.

The velocity update rule of DNB-FPSO uses the formula given below while the position update rule is the same as the position update rule of original PSO (i.e. Eq. (6)):

$$v_i^d(t+1) = w(t) \times v_i^d(t) + c_1 \times r_1 \times (pbest_i^d(t) - x_i^d(t)) + c_2 \times r_2 \times (DNB_i^d(t) - x_i^d(t)), \tag{8}$$

where $\overrightarrow{DNB}_i(t) = (DNB_i^1(t), \ldots, DNB_i^d(t), \ldots, DNB_i^n(t))$ is the DNB personal best position over the neighborhood of the personal best of particle i is obtained by Algorithm (3). The time complexity of Algorithm (3) is $O(n \times N^2)$. Note

that the lower bound of the swarm and evolutionary algorithms that calculate the distance matrix between each pair of the swarm in decision space is $O(n \times N^2)$. Therefore, the DNB neighborhood is an efficient distance-based neighborhood in decision space, because based on the theory of computational complexity it does not add any significant overhead to the algorithm. For example, the swarm and evolutionary algorithms that try to find K-nearest neighbors of each particle in decision space are not efficient, because their time complexity is $O(K \times n \times N^2)$. The pseudo-code of the

Algorithm (3): Outline of finding all DNB neighbors for minimization problems.

Inputs: Personal best position of all particles.
for $i = 1$ *to* N **do**
  $DNB_i(t) = 0$;
  $minDist = \infty$;
  for $j = 1$ *to* N **do**
    if $f(\overrightarrow{Pbest}_j(t)) < f(\overrightarrow{Pbest}_i(t))$ ***AND*** *Distance* $(\overrightarrow{Pbest}_i(t), \overrightarrow{Pbest}_j(t)) < minDist$ **then**
      $DNB_i(t) = j$;
      minDist=Distance $(\overrightarrow{Pbest}_i(t), \overrightarrow{Pbest}_j(t))$;
    end
  end
  if $DNB_i(t) == 0$ **then**
    $DNB_i(t) = i$;
  end
end
Output: $\overrightarrow{DNB}(t) = (DNB_1, DNB_2, \ldots, DNB_N)$.

DNB-FPSO is described in Algorithm (4).

Algorithm (4): Outline of DNB-FPSO for minimization problems.

Initialize w, N, and stopping criterion;
$t = 0$;
for $i = 1$ *to* N **do**
  Randomly generate the initial solution $\vec{X}_i(t)$;
  $\overrightarrow{Pbest}_i(t) = \vec{X}_i(t)$;
  Randomly generate the initial velocity $\vec{V}_i(t)$;
end
while *stopping criterion is not satisfied* **do**
  Calculate $\overrightarrow{DNB}(t) = \{DNB_1, DNB_2, \ldots, DNB_N\}$ by Algorithm (3);
  for $i = 1$ *to* N **do**
    for $d = 1$ *to* N **do**
      Calculate the next velocity $v_i^d(t+1)$ by Eq. (8);
      Update the next position $x_i^d(t+1)$;
    end
    if $f(\vec{x}_i(t+1)) < f(\overrightarrow{Pbest}_i(t))$ **then**
      $\overrightarrow{Pbest}_i(t+1) = \vec{x}_i(t+1)$;
    else
      $\overrightarrow{Pbest}_i(t+1) = \overrightarrow{Pbest}_(t)$;
    end
  end
  $t = t + 1$;
  Update $w(t)$ by fuzzy balancer;
end
Output: $\vec{x}_i(t), i = 1, \ldots, N$.

3.3 Fuzzy balancer

In this section, a fuzzy inference system is proposed to intelligently manage the balance between exploration and exploitation by adjusting the value of $w(t)$. For this purpose, a zero-order Sugeno fuzzy inference system with two inputs and one output is used [64]. Two inputs for the proposed fuzzy inference system are the normalized current iteration of the algorithm and the normalized diversity of swarm in decision space. The reason for selecting the current iteration as the first input is clear: each algorithm must explore the search space with a diverse swarm at the beginning and change into convergence by lapse of iterations. In this paper, the normalized current iteration is calculated for fuzzy balancer as follows:

$$iter = \frac{t}{T}, \tag{9}$$

where t is the number of iterations elapsed and T is the maximal number of iterations. It is necessary to mention that *iter* is equal to "0" if the algorithm is in the first iteration, is equal to "1" if the algorithm is in the last iteration, and in general we have $0 \leqslant iter \leqslant 1$. The main reason for selecting the diversity of swarm as a second input of fuzzy balancer is that whatever the swarm diversity is, we have good exploration and it does not require more exploration. In this paper, we propose a new equation to calculate diversity as follows:

$$\text{diversity} = \frac{4}{n \times (b-a)^2} \sum_{d=1}^{n} \text{variance}(x_1^d, x_2^d, \ldots, x_N^d), \tag{10}$$

where N is the number of particles, n is the number of dimensions, b and a are upper and lower bounds of variables, and $variance(x_1^d, x_2^d, \ldots, x_N^d)$ is the variance between all particles in the dimension d. It is noteworthy that *diversity* is equal to "0" if all particles converge to a similar solution, is equal to "1" if for each dimension of all particles we have $variance(x_1^d, x_2^d, \ldots, x_N^d) = \frac{(b-a)^2}{4}$ in which $\frac{(b-a)^2}{4}$ is the upper bound on the variance of each variable that takes on values in range $[a, b]$. Therefore, in general we have $0 \leqslant \text{diversity} \leqslant 1$. The output of the proposed fuzzy inference system is the value of $w(t)$, because the exploration ability of the proposed algorithm depends on its value. As we know, a larger value of this parameter encourages exploration, while a smaller value of it provides exploitation. Using these facts, the following rule-base is proposed to calculate the value of $w(t)$ by the fuzzy inference system:

1. If *iter* is *Low* and *diversity* is *Low*, then $w(t) = VHigh$.
2. If *iter* is *Low* and *diversity* is *Med*, then $w(t) = High$.
3. If *iter* is *Low* and *diversity* is *High*, then $w(t) = Med$.
4. If *iter* is *Med* and *diversity* is *Low*, then $w(t) = High$.
5. If *iter* is *Med* and *diversity* is *Med*, then $w(t) = Med$.
6. If *iter* is *Med* and *diversity* is *High*, then $w(t) = Low$.
7. If *iter* is *High* and *diversity* is *Low*, then $w(t) = Med$.
8. If *iter* is *High* and *diversity* is *Med*, then $w(t) = Low$.
9. If *iter* is *High* and *diversity* is *High*, then $w(t) = VLow$.

For partitioning the input spaces of iter and *diversity* variables, we use a general rule where the resulting fuzzy term sets for both variables are orthogonal with ϵ-completeness equal to 0.5. Fig. 3 shows the input membership functions of the proposed fuzzy inference system. The rules consequences are set as: $VHigh = 1, High = 0.5, Med = 0.2, Low = 0.1, and VLow = 0.01$.

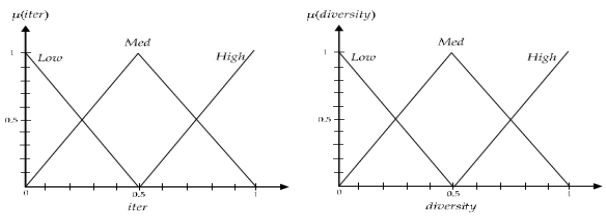

Fig 3. Input membership functions of the proposed fuzzy inference system. Two inputs for the proposed fuzzy inference system are *iter* (the normalized current iteration of the algorithm) and *diversity* (the normalized diversity of swarm in decision space).

4 Numerical experiments

4.1 Experimental setup

In the following, we first describe the characteristics of the selected benchmarks. Then, the two performance indicators used in this paper are presented. Finally, the experimental setting of TNB-FPSO and DNB-FPSO are described.

4.1.1 Benchmarks description

In this paper, eight novel standard benchmark multimodal test functions [47] are used to evaluate the performance of the proposed algorithms. Table 1 lists the basic information of these test functions which coordinate rotation and shift operations to create linkage among different dimensions and to place the optima at a different location, respectively [47]. Note that all these benchmarks are minimization functions.

Table 1
Main characteristics of the eight expanded and composition minimization multimodal functions [47].

Test Function Name	Dimensions	No. of Global/Local Peaks Peaks	Optimum Value
F1: Shifted and rotated expanded two-peak trap	5	1/15	100
	10	1/55	100
	20	1/210	100
F2: Shifted and rotated expanded Five-Uneven-Peak Trap	2	4/21	200
	5	32/0	200
	8	256/0	200
F3: Shifted and rotated Expanded Equal Minima	2	25	300
	3	125/0	300
	4	625/0	300
F4: Shifted and rotated Expanded Decreasing Minima	5	1/15	400
	10	1/55	400
	20	1/210	400
F5: Shifted and rotated expanded uneven minima	2	25/0	500
	3	125/0	500
	4	625/0	500
F6: Shifted and rotated expanded Himmelblaus function	4	16/0	600
	6	64/0	600
	8	256/0	600
F7: Shifted and rotated expanded six-hump camel back	6	8/0	700
	10	32/0	700
	16	256/0	700
F8: Shifted and rotated modified Vincent function	2	36/0	800
	3	216/0	800
	4	1296/0	800

4.1.2 Performance measure

In this paper, Average Number of Optima Found (ANOF) as performance indicator are used which use the locations of the optima. ANOF use the given level of accuracy, which a level of accuracy, typically $0 < \epsilon < 1$, is a value indicating how close the computed solutions to the known global peaks are. If the difference from a computed solution to a known global optimum is below ϵ, then the peak is considered to have been found. Moreover, to ensure different peaks are located by different solutions, a distance restriction is also imposed. That is only when the distance between one solution and its respective optimum is less than $dis_{threshold}$ and the difference is below ϵ, the true solution is considered to have been found. In this way, it avoids one solution being counted more than once when there is no solution near an optimum.

4.1.3 Experimental setting

In total, seven multimodal optimization algorithms are examined in our experiments:

- TNB-FPSO: the proposed Topologically Nearest-Better Fuzzy PSO,
- DNB-FPSO: the proposed Distance-based Nearest-Better Fuzzy PSO,
- LIPS [48]: Locally Informed PSO,
- R2PSO [35]: a PSO with a ring topology; each particle of swarm interacts only with its immediate particle to its right.
- FER-PSO [34]: PSO based on Fitness Euclidean-distance Ratio.
- MOMMOP [63]: Multi-objective Optimization for locating multiple optimal solutions of Multimodal Optimization Problems.
- TriDEMO [66]: Tri-objective Differential Evolution for multimodal optimization.

In this experiment, the level of accuracy (ϵ), niching radius ($dis_{threshold}$, population size, and maximal number of function evaluations allowed are as follows (these settings are applied to all compared algorithms):

- the level of accuracy (ϵ) is set by 0.001,
- the niching radius is set by 0.1*n where n is the dimension of the problem,
- the population size is set by 500 * $Round(\sqrt{n})$, and
- the maximal number of function evaluations is set by 2000 * n * $\sqrt{q}$ where q is the number of optima.

Generally, population size and maximal number of function evaluations are related to the number of optima to be located. Large number of optima requires a larger population size and more function evaluations. Note that the performance measure of each algorithm depends on the specified level of accuracy.

To proposed TNB-FPSO and DNB-FPSO, we set $c_1 = 1$ and $c_2 = 2$. The values of these parameters are obtained experimentally. It is noteworthy that the experimental settings of LIPS, R2PSO, FER-PSO, MOMMOP, and TriDEMO are adopted exactly as in their original works. All algorithms were implemented in MATLAB 2015a environment and run on a PC with an Intel 2.2 GHz CPU. Each algorithm is run for 50 independent replications.

4.2 Experimental results

In this section, we evaluate the effectiveness of the TNB-FPSO and DNB-FPSO on the several complex multimodal optimization functions. Table 3 illustrates the results of TNB-FPSO, DNB-FPSO, and four recent multimodal optimization algorithms LIPS, R2PSO, FER-PSO, MOMMOP, and TriDEMO in case of ANOF for F1-F8.

In Table 3 the best algorithm is highlighted in boldface for each test function. As can be seen, DNB-FPSO performs the best on the most test functions. It can be seen that DNB-PSO performs the best on most of the test functions (13 out of 24 test functions). DNB-PSO also shows clear superiority compared with each of the six multimodal optimization algorithms. In order to determine the statistical significance of the advantage of TNB-FPSO and DNB-FPSO, t-test (all compared with TNB-FPSO and all compared with DNB-FPSO) is applied and shown in two last rows of each test function. The symbols $+$, $\approx$, and $-$ represent that other methods are statistically inferior to, equal to, or superior to the proposed algorithm, respectively. The last six rows of this table summarize how many cases that TNB-FPSO and DNB-FPSO perform better, similar, or worse than other algorithms. From the results, we can observe that DNB-FPSO always perform better or equal when compared with LIPS, R2PSO, FER-PSO, and MOMMOP methods on the test functions F1F8.

Figs. 4 and 5 plot the convergence behavior and quality attained by TNB-FPSO, DNB-FPSO, LIPS, R2PSO, FER-PSO, and MOMMOP over function evaluation for functions F3 and F8 of Table 1, respectively. As can be seen, the proposed algorithms have better convergence performance and obtain better results than that of the other multimodal optimization algorithms. The above statistical comparisons confirm the effectiveness of Topological Nearest-Better (TNB) neighborhood and Distance-based Nearest-Better (DNB) neighborhood when used together with the swarm intelligence algorithms.

Table 2

Comparison of TNB-FPSO, DNB-FPSO, LIPS, R2PSO, FER-PSO, MOMMOP, and TriDEMO in case of ANOF for expanded minimization multimodal functions F1-F8.

Function No.	Dimension	Measure	TNB-FPSO	DNB-FPSO	LIPS	R2PSO	FER-PSO	MOMMOP	TriDEMO
F1	5	Mean	0	**1.74**	0	0	0	0	1.00
		Std	0	0.80	0	0	0	0	0.60
		t-test (TNB-FPSO)	NA	-	≈	≈	≈	≈	-
		t-test (DNB-FPSO)	+	NA	+	+	+	+	≈
	10	Mean	0	0	0	0	0	0	0
		Std	0	0	0	0	0	0	0
		t-test (TNB-FPSO)	NA	≈	≈	≈	≈	≈	≈
		t-test (DNB-FPSO)	≈	NA	≈	≈	≈	≈	≈
	20	Mean	0	0	0	0	0	0	0
		Std	0	0	0	0	0	0	0
		t-test (TNB-FPSO)	NA	≈	≈	≈	≈	≈	≈
		t-test (DNB-FPSO)	≈	NA	≈	≈	≈	≈	≈
F2	2	Mean	3.65	**4.56**	3.12	3.57	1.85	0.42	3.69
		Std	0.47	0.65	0.49	0.62	0.97	0.73	0.59
		t-test (TNB-FPSO)	NA	≈	≈	≈	+	+	≈
		t-test (DNB-FPSO)	≈	NA	≈	≈	+	+	≈
	5	Mean	0	1.14	0	0	0	0	**9.04**
		Std	0	0.68	0	0	0	0	2.59
		t-test (TNB-FPSO)	NA	-	≈	≈	≈	≈	-
		t-test (DNB-FPSO)	+	NA	+	+	+	+	-
	8	Mean	0	10.50	0	0	0	0	**18.75**
		Std	0	2.63	0	0	0	0	3.73
		t-test (TNB-FPSO)	NA	-	≈	≈	≈	≈	-
		t-test (DNB-FPSO)	+	NA	+	+	+	+	-
F3	2	Mean	22.83	**23.70**	19.64	19.03	22.12	20.27	19.63
		Std	1.05	0.97	1.49	1.28	1.37	2.15	1.84
		t-test (TNB-FPSO)	NA	≈	+	+	≈	+	+
		t-test (DNB-FPSO)	≈	NA	+	+	≈	+	+
	3	Mean	**64.23**	54.06	20.84	7.39	14.46	16.71	49.36
		Std	3.04	3.27	4.21	2.37	2.75	4.26	4.02
		t-test (TNB-FPSO)	NA	+	+	+	+	+	+
		t-test (DNB-FPSO)	-	NA	+	+	+	+	+
	4	Mean	69.37	60.26	25.17	2.73	17.27	7.47	**80.71**
		Std	4.42	5.28	2.84	0.84	4.05	1.37	7.42
		t-test (TNB-FPSO)	NA	+	+	+	+	+	-
		t-test (DNB-FPSO)	-	NA	+	+	+	+	-
F4	5	Mean	0.65	**1.06**	0	0	0	0	0.95
		Std	0.27	0.32	0	0	0	0	0.60
		t-test (TNB-FPSO)	NA	-	+	+	+	+	≈
		t-test (DNB-FPSO)	+	NA	+	+	+	+	≈
	10	Mean	0	0	0	0	0	0	0
		Std	0	0	0	0	0	0	0
		t-test (TNB-FPSO)	NA	≈	≈	≈	≈	≈	≈
		t-test (DNB-FPSO)	≈	NA	≈	≈	≈	≈	≈
	20	Mean	0	0	0	0	0	0	0
		Std	0	0	0	0	0	0	0
		t-test (TNB-FPSO)	NA	≈	≈	≈	≈	≈	≈
		t-test (DNB-FPSO)	≈	NA	≈	≈	≈	≈	≈
F5	2	Mean	22.21	**24.42**	17.02	15.85	20.06	22.05	21.74
		Std	1.70	2.82	1.06	2.18	1.46	1.47	1.87
		t-test (TNB-FPSO)	NA	≈	+	+	+	≈	≈
		t-test (DNB-FPSO)	≈	NA	+	+	+	+	+
	3	Mean	**62.25**	55.38	27.35	0	15.06	10.82	42.24
		Std	5.31	6.08	3.72	0	3.42	1.95	3.90
		t-test (TNB-FPSO)	NA	+	+	+	+	+	+
		t-test (DNB-FPSO)	-	NA	+	+	+	+	+
	4	Mean	**73.20**	62.64	23.80	0	17.54	0	53.47
		Std	5.22	5.39	5.49	0	2.80	0	4.52
		t-test (TNB-FPSO)	NA	+	+	+	+	+	+
		t-test (DNB-FPSO)	-	NA	+	+	+	+	+

Function No.	Dimension	Measure	TNB-FPSO	DNB-FPSO	LIPS	R2PSO	FER-PSO	MOMMOP	TriDEMO
F6	4	Mean	7.30	**11.76**	0.50	0.04	0	0	11.23
		Std	0.94	1.26	0.71	0.2	0	0	1.85
		t-test (TNB-FPSO)	NA	-	+	+	+	+	-
		t-test (DNB-FPSO)	+	NA	+	+	+	+	≈
	6	Mean	4.50	**44.88**	40.10	0	0	0	10.97
		Std	1.90	1.66	1.97	0	0	0	1.64
		t-test (TNB-FPSO)	NA	-	-	+	+	+	-
		t-test (DNB-FPSO)	+	NA	≈	+	+	+	+
	8	Mean	0.80	**81.20**	65.70	0	0	0	7.32
		Std	0.78	5.18	2.91	0	0	0	1.05
		t-test (TNB-FPSO)	NA	-	-	+	+	+	-
		t-test (DNB-FPSO)	+	NA	+	+	+	+	+
F7	6	Mean	**2.90**	1.8	0	0	0	0	1.48
		Std	1.52	0.80	0	0	0	0	0.73
		t-test (TNB-FPSO)	NA	+	+	+	+	+	+
		t-test (DNB-FPSO)	-	NA	+	+	+	+	≈
	10	Mean	3.10	**6.32**	0	0	0	0	2.95
		Std	1.66	2.01	0	0	0	0	1.57
		t-test (TNB-FPSO)	NA	-	+	+	+	+	≈
		t-test (DNB-FPSO)	+	NA	+	+	+	+	+
	16	Mean	1.60	**25.52**	0.10	0	0	0	2.63
		t-test (TNB-FPSO)	NA	-	+	+	+	+	≈
		t-test (DNB-FPSO)	+	NA	+	+	+	+	+
F8	2	Mean	17.40	**32.12**	29.72	17.52	27.92	10.30	18.16
		Std	2.54	1.50	1.91	2.84	1.84	2.05	2.94
		t-test (TNB-FPSO)	NA	-	-	≈	-	+	≈
		t-test (DNB-FPSO)	+	NA	≈	+	≈	+	+
	3	Mean	34.90	**88.60**	75.04	35.76	59.80	2.30	55.71
		Std	3.60	4.70	4.48	5.92	3.97	1.70	3.82
		t-test (TNB-FPSO)	NA	-	-	≈	-	+	-
		t-test (DNB-FPSO)	+	NA	+	+	+	+	+
	4	Mean	34.40	**121.92**	90.20	1.24	72.96	0.8	63.47
		Std	3.56	5.04	6.05	1.12	5.64	0.94	5.16
		t-test (TNB-FPSO)	NA	-	-	+	-	+	-
		t-test (DNB-FPSO)	+	NA	+	+	+	+	+
t-test summary for TNB-FPSO against others		Better	NA	5	11	14	13	16	5
		Similar	NA	7	8	10	8	8	10
		Worse	NA	12	5	0	3	0	9
t-test summary for DNB-FPSO against others		Better	12	NA	17	19	18	20	13
		Similar	7	NA	7	5	6	4	8
		Worse	5	NA	0	0	0	0	3

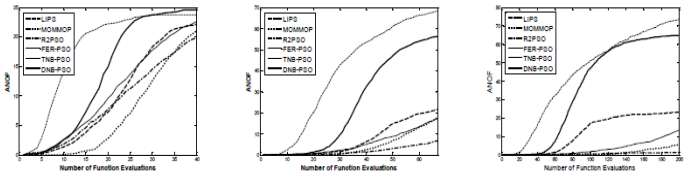

Fig 4. Convergence behavior and quality attained by TNB-FPSO, DNB-FPSO, LIPS, R2PSO, FER-PSO, and MOMMOP over function evaluation for function F3 with: (a) 2 dimensions, (b) 3 dimensions, and (c) 4 dimensions. These results indicate that the proposed methods generates better solution set than the other multimodal algorithms for function F3.

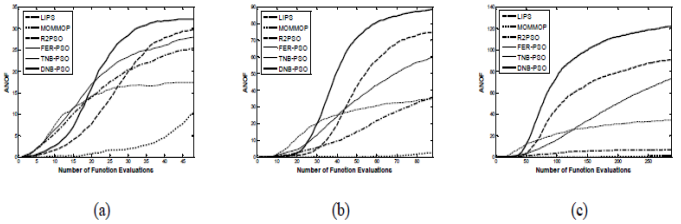

Fig 5. Convergence behavior and quality attained by TNB-FPSO, DNB-FPSO, LIPS, R2PSO, FER-PSO, and MOMMOP over function evaluation for function F8 with: (a) 2 dimensions, (b) 3 dimensions, and (c) 4 dimensions.

These results indicate that the proposed methods generates better solution set than the other multimodal algorithms for function F3.

4.3 Discussion

In this section the features of TNB-FPSO and DNB-FPSO are compared with each other and other multimodal optimization algorithms, respectively.

4.3.1 TNB-FPSO vs. DNB-FPSO

DNB-FPSO is more effective than TNB-FPSO. In the TNB-FPSO, it cannot be avoided that some particles must have topological neighbors from different niches. If two particles are from different niches, they may oscillate between two peaks that waste the function evaluations. During the exploration stage (early search stage), particles are confined to locate additional niches that are positioned between two previously identified niches due to the oscillation. However, if there is no niche between two previously identified niches, oscillation wastes function evaluations. In contrast to TNB-FPSO, DNB-FPSO significantly reduces the oscillation between particles, because in most cases a particle and its nearest-better neighbor are within the same niche.

TNB-FPSO is more efficient than DNB-FPSO. The time complexity of TNB-FPSO is $O(N)$, whereas the time complexity of DNB-FPSO is $O(n \times N^2)$. The lower efficiency of DNB-FPSO is due to the computation of the distance matrix of particles in decision space. The distance calculation can be expensive since every particle has to be compared with all others in decision space.

4.3.2 Nearest-better neighborhood vs. other popular neighborhoods

Nearest-Better neighborhood is a parameter-less technique. In contrast to the most neighborhood, nearest-better neighborhood does not require specification of any explicitly niching parameters such as how far apart between two optima or the number of optima. For example, there exist a niching parameter in LIPS, i.e. the neighborhood size parameter, which must be set by a user. In FER-PSO, the FER value is calculated based on the ratio of the fitness difference and the Euclidean distance between a solutions personal best and other personal best of the solutions in the population, scaled by a factor so that neither the fitness nor the Euclidean distance becomes too dominating. The calculation of FER would require some knowledge about the search range of each variable. It is obvious that this knowledge may not be available in many real-world optimization problems.

Nearest-Better neighborhood is an efficient technique. In the best of our knowledge, the time complexity of existing best topological and distance-based PSO algorithms are $O(N)$ and $O(n \times N^2)$, respectively. Hence, we can say that nearest-better neighborhood is an efficient niching technique. For a better comparison, Table 4 shows the growth order of the time complexity for the proposed algorithms and other PSO algorithms.

Table 4
The growth order of the time complexity for the proposed algorithms and other PSO algorithms.

Topological PSO algorithms		Distance-based PSO algorithms	
Algorithm	Complexity	Algorithm	Complexity
RPSO	$O(N)$	FER-PSO	$O(n \times N^2)$
TNB-FPSO	$O(N)$	LIPS	$O(nsize \times n \times N^2)$
		DNB-FPSO	$O(n \times N^2)$

5 Conclusions and future works

Most existing Particle Swarm Optimization (PSO) require specification of some niching parameters to solve multimodal optimization problems. In this paper, we proposed two novel parameter-less niching versions of PSO, called Topologically Nearest-Better Fuzzy PSO (TNB-FPSO) and Distance-based Nearest-Better Fuzzy PSO (DNB-FPSO), for solving multimodal optimization problems. It should be noted that we proposed a zero-order fuzzy system to balance between exploration and exploitation in the proposed algorithms. The proposed algorithms do not require specification of any niching parameters such as how far apart between two optima, or the number of optima. Empirical evaluation of the TNB-FPSO and DNB-FPSO on the several complex multimodal benchmark functions demonstrate that proposed algorithms can obtain better results than the other compared multimodal optimization algorithms in the terms of final solution set quality. Therefore, this paper presents two versions of the FPSO that remove undesirable parameters of niching without sacrificing performance. Comparison between the TNB-FPSO and DNB-FPSO shows that the TNB-FPSO has a better efficiency, but Algorithm 2 has a better effectiveness.

Although TNB-FPSO is more efficient than DNB-FPSO, it has one major problem: If two particles are from different niches, they may oscillate between two peaks that waste the function evaluations. In contrast to TNB-FPSO, DNB-FPSO significantly reduces the oscillation between particles, because in most cases a particle and its nearest-better neighbor are within the same niche. Therefore, a first step toward extending this paper would be to fix the problem of particle oscillations in TNB-FPSO. Secondly, the nearest-better neighborhood can be used in other swarm intelligence algorithms such as Competitive Swarm Optimization (CSO), Gravitational Search Algorithm (GSA), and Ant Colony Optimization (ACO). Finally, the nearest-better neighborhood can be adapted for solving multi-objective and many-objective optimization problems.

References

[1] A. Bienvenue, M. Joannides, J. Berard, E. Fontenas, O. Francois, *Niching in monte carlo filtering algorithms*, in Artificial Evolution, ser. Lecture Notes in Computer Science, P. Collet, C. Fonlupt, J. K. Hao, E. Lutton, and M. Schoenauer, Eds. Springer Berlin Heidelberg, (2013), 19-30.

[2] S. Bird, X. Li, *Adaptively choosing niching parameters in a PSO*, In Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, (2006, July), 3-10. ACM.

[3] S. Bird, X. Li, *Enhancing the robustness of a speciation-based PSO*, In 2006, IEEE International Conference on Evolutionary Computation, (2006, July), 843-850.

[4] C. L. Blake, C. J. Merz, *UCI repository of machine learning databases*, Available from: http://www.ics.uci.edu/ mlearn/MLRepository.html.

[5] R. Brits, A. P. Engelbrecht, F. Van den Bergh, *A niching particle swarm optimizer*, In Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning, **2** (2002, November), 692-696.

[6] D. Chang, Y. Zhao, Y. Xiao, *A robust dynamic niching genetic clustering approach for image segmentation*, In Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, ser. GECCO 11. New York, NY, USA: ACM, (2011), 1077-1084.

[7] N. Cheikhrouhou, B. Sarkar, B. Ganguly, A. I. Malik, R. Batista, Y. H. Lee, *Optimization of sample size and order size in an inventory model with quality inspection and return of defective items*, Annals of Operations Research, **271**(2) (2018), 445-467.

[8] D. H. Cho, H. K. Jung, C. G. Lee, *Induction motor design for electric vehicle using a niching genetic algorithm,* IEEE Transactions on Industry Applications, **37**(4) (2001), 994-999.

[9] C. Cobos, C. Montealegre, M. F. Mejia, M. Mendoza, E. Leon, *Web document clustering based on a new niching memetic algorithm, term-document matrix and bayesian information criterion,* in IEEE Congress on Evolutionary Computation, (2010), 18.

[10] K. Delibasis, P. A. Asvestas, G. K. Matsopoulos, *Multimodal genetic algorithms-based algorithm for automatic point correspondence,* Pattern Recognition, **43**(12) (2010), 4011-4027.

[11] M. B. Dowlatshahi, V. Derhami, *Winner determination in combinatorial auctions using hybrid ant colony optimization and multi-neighborhood local search*, Journal of AI and Data Mining, **5**(2) (2017), 169-181.

[12] M. B. Dowlatshahi, V. Derhami, H. Nezamabadi-pour, *Ensemble of filter-based rankers to guide an epsilon-greedy swarm optimizer for high-dimensional feature subset selection*, Information, **8**(4) (2017), 152.

[13] M. B. Dowlatshahi, V. Derhami, H. Nezamabadi-pour, *A novel three-stage filter-wrapper framework for miRNA subset selection in cancer classification*, Informatics, **5**(1) (2018), 13.

[14] M. B. Dowlatshahi, H. Nezamabadi-Pour, *GGSA: A grouping gravitational search algorithm for data clustering*, Engineering Applications of Artificial Intelligence, **36** (2014), 114-121.

[15] M. B. Dowlatshahi, H. Nezamabadi-Pour, M. Mashinchi, *A discrete gravitational search algorithm for solving combinatorial optimization problems*, Information Sciences, **258** (2014), 94-107.

[16] M. B. Dowlatshahi, M. Rezaeian, *Training spiking neurons with gravitational search algorithm for data classification*, In 2016 1st Conference on Swarm Intelligence and Evolutionary Computation (CSIEC), (2016), 53-58.

[17] H. Garg, *Fuzzy multiobjective reliability optimization problem of industrial systems using particle swarm optimization*, Journal of Industrial Mathematics, (2013), 1-9.

[18] H. Garg, *An approach for analyzing fuzzy system reliability using particle swarm optimization and intuitionistic fuzzy set theory*, Journal of Multiple-Valued Logic and Soft Computing, **21**(3) (2013), 21.

[19] H. Garg, *A hybrid PSO-GA algorithm for constrained optimization problems*, Applied Mathematics and Computation, **274** (2016), 292-305.

[20] H. Garg, *A hybrid GSA-GA algorithm for constrained optimization problems*, Information Sciences, **478** (2019), 499-523.

[21] H. Garg, S. P. Sharma, *Multi-objective reliability-redundancy allocation problem using particle swarm optimization*, Computers and Industrial Engineering, **64**(1) (2013), 247-255.

[22] Y. J. Gong, J. Zhang, Y. Zhou, *Learning multimodal parameters: A bare-bones niching differential evolution approach*, IEEE Transactions on Neural Networks and Learning Systems, **29**(7) (2017), 2944-2959.

[23] L. Huang, C. T. Ng, A. H. Sheikh, M. C. Griffith, *Niching particle swarm optimization techniques for multimodal buckling maximization of composite laminates*, Applied Soft Computing, **57** (2017), 495-503.

[24] M. Iwamatsu, *Multi-species particle swarm optimizer for multimodal function optimization*, IEICE Transactions on Information and Systems, **89**(3) (2006), 1181-1187.

[25] P. Jiang, X. Liu, C. A. Shoemaker, *An adaptive particle swarm algorithm for unconstrained global optimization of multimodal functions*, In Proceedings of the 9th International Conference on Machine Learning and Computing, (2017), 221-226.

[26] S. Jiang, Y. S. Ong, J. Zhang, L. Feng, *Consistencies and contradictions of performance metrics in multiobjective optimization*, IEEE Transactions on Cybernetics, **44**(12) (2014), 2391-2404.

[27] S. Kamyab, M. Eftekhari, *Feature selection using multimodal optimization techniques*, Neurocomputing, **171** (2016), 586-597.

[28] C. W. Kang, M. Imran, M. Omair, W. Ahmed, M. Ullah, B. Sarkar, *Stochastic-petri net modeling and optimization for outdoor patients in building sustainable healthcare system considering staff absenteeism*, Mathematics, **7**(6) (2019), 499.

[29] J. Kennedy, *Particle swarm optimization*, in Encyclopedia of Machine Learning, Springer, Boston, MA, (2010), 760-766.

[30] J. Kennedy, R. Eberhart, *Particle swarm optimization*, In: Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, (1995), 1942-1948.

[31] K. D. Koper, M. E. Wysession, D. A. Wiens, *Multimodal function optimization with a niching genetic algorithm: A seismological example*, Bulletin of the Seismological Society of America, **89**(4) (1999), 978-988.

[32] M. Kronfeld, A. Dräger, M. Aschoff, A. Zell, *On the benefits of multimodal optimization for metabolic network modeling*, In German Conference on Bioinformatics 2009, Gesellschaft Für Informatik eV, 2009.

[33] X. Li, *Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization*, In Genetic and Evolutionary Computation Conference, Springer, Berlin, Heidelberg, (2004), 105-116.

[34] X. Li, *A multimodal particle swarm optimizer based on fitness Euclidean-distance ratio*, In Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, (2007), 78-85.

[35] X. Li, *Niching without niching parameters: Particle swarm optimization using a ring topology*, IEEE Transactions on Evolutionary Computation, **14**(1) (2009), 150-169.

[36] X. Li, A. Engelbrecht, M. G. Epitropakis, *Benchmark functions for CEC2013 special session and competition on niching methods for multimodal function optimization*, RMIT University, Evolutionary Computation and Machine Learning Group, Australia, Tech. Rep, 2013.

[37] X. Li, M. G. Epitropakis, K. Deb, A. Engelbrecht, *Seeking multiple solutions: An updated survey on niching methods and their applications*, IEEE Transactions on Evolutionary Computation, **21**(4) (2016), 518-538.

[38] Y. Liu, X. Ling, Z. Shi, M. Lv, J. Fang, L. Zhang, *A survey on particle swarm optimization algorithms for multimodal function optimization*, Journal of Software, **6**(12) (2011), 2449-2455.

[39] E. Özcan, M. Yílmaz, *Particle swarms for multimodal optimization*, In International Conference on Adaptive and Natural Computing Algorithms, Springer, Berlin, Heidelberg, (2007), 366-375.

[40] R. S. Patwal, N. Narang, H. Garg, *A novel TVAC-PSO based mutation strategies algorithm for generation scheduling of pumped storage hydrothermal system incorporating solar units*, Energy, **142** (2018), 822-837.

[41] A. Passaro, A. Starita, *Particle swarm optimization for multimodal functions: A clustering approach*, Journal of Artificial Evolution and Applications, (2008), 1-15.

[42] E. Pérez, F. Herrera, C. Hernández, *Finding multiple solutions in job shop scheduling by niching genetic algorithms*, Journal of Intelligent Manufacturing, **14**(3-4) (2003), 323-339.

[43] E. Pérez, M. Posada, F. Herrera, *Analysis of new niching genetic algorithms for finding multiple solutions in the job shop scheduling,* Journal of Intelligent Manufacturing, **23**(3) (2012), 341-356.

[44] E. Pérez, M. Posada, A. Lorenzana, *Taking advantage of solving the resource constrained multi-project scheduling problems using multi-modal genetic algorithms*, Soft Computing, **20**(5) (2016), 1879-1896.

[45] M. Preuss, *Multimodal optimization by means of evolutionary algorithms*, Springer International Publishing, 2015.

[46] M. Preuss, L. Schönemann, M. Emmerich, *Counteracting genetic drift and disruptive recombination in* $(\mu + \lambda)$*-EA on multimodal fitness landscapes*, In Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, (2005), 865-872.

[47] B. Y. Qu, J. J. Liang, Z. Y. Wang, Q. Chen, P. N. Suganthan, *Novel benchmark functions for continuous multimodal optimization with comparative results*, Swarm and Evolutionary Computation, **26** (2016), 23-34.

[48] B. Y. Qu, P. N. Suganthan, S. Das, *A distance-based locally informed particle swarm model for multimodal optimization*, IEEE Transactions on Evolutionary Computation, **17**(3) (2012), 387-402.

[49] M. K. Rafsanjania, M. B. Dowlatshahi, H. Nezamabadi-Pour, *Gravitational search algorithm to solve the K-of-N lifetime problem in two-tiered WSNs*, Iranian Journal of Mathematical Sciences and Informatics, **10**(1) (2015), 81-93.

[50] J. L. Redondo, J. Fernández, I. GarcÍa, P. M. Ortigosa, *Solving the multiple competitive facilities location and design problem on the plane*, Evolutionary Computation, **17**(1) (2009), 21-53.

[51] D. E. Rumelhart, G. E. Hinton, R. J. Williams, *Learning representations by back-propagating errors*, Cognitive Modeling, **5**(3) (1988), 1.

[52] R. Sahajpal, G. V. Ramaraju, V. Bhatt, *Applying niching genetic algorithms for multiple cluster discovery in spatial analysis*, In International Conference on Intelligent Sensing and Information Processing, Proceedings, (2004), 35-40.

[53] B. Sarkar, *Mathematical and analytical approach for the management of defective items in a multi-stage production system*, Journal of Cleaner Production, **218** (2019), 896-919.

[54] B. Sarkar, A. Majumder, M. Sarkar, B. K. Dey, G. Roy, *Two-echelon supply chain model with manufacturing quality improvement and setup cost reduction*, Journal of Industrial and Management Optimization, **13**(2) (2017), 1085-1104.

[55] B. Sarkar, M. Omair, S. B. Choi, *A multi-objective optimization of energy, economic, and carbon emission in a production model under sustainable supply chain management*, Applied Sciences, **8**(10) (2018), 1744.

[56] B. Sarkar, B. K. Shaw, T. Kim, M. Sarkar, D. Shin, *An integrated inventory model with variable transportation cost, two-stage inspection, and defective items*, Journal of Industrial and Management Optimization, **13**(4) (2017), 1975-1990.

[57] T. Sasaki, H. Nakano, A. Miyauchi, A. Taguchi, *Particle swarm optimizer networks with stochastic connection for improvement of diversity search ability to solve multimodal optimization problems*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, **100**(4) (2017), 996-1007.

[58] J. H. Seo, C. H. Im, C. G. Heo, J. K. Kim, H. K. Jung, C. G. Lee, *Multimodal function optimization based on particle swarm optimization*, IEEE Transactions on Magnetics, **42**(4) (2006), 1095-1098.

[59] J. H. Seo, C. H. Im, S. Y. Kwak, *An improved particle swarm optimization algorithm mimicking territorial dispute between groups for multimodal function optimization problems,* IEEE Transactions on Magnetics, (2008), 1046-1049.

[60] O. M. Shir, C. Siedschlag, T. Bäck, M. J. Vrakking, *Niching in evolution strategies and its application to laser pulse shaping*, In International Conference on Artificial Evolution (Evolution Artificielle), Springer, Berlin, Heidelberg, (2005), 85-96.

[61] W. Song, Y. Wang, H. X. Li, Z. Cai, *Locating multiple optimal solutions of nonlinear equation systems based on multiobjective optimization*, IEEE Transactions on Evolutionary Computation, **19**(3) (2014), 414-431.

[62] A. A. Taleizadeh, H. Samimi, B. Sarkar, B. Mohammadi, *Stochastic machine breakdown and discrete delivery in an imperfect inventory-production system*, Journal of Industrial and Management Optimization, **13**(3) (2017), 1511-1535.

[63] Y. Wang, H. X. Li, G. G. Yen, W. Song, *MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems*, IEEE Transactions on Cybernetics, **45**(4) (2014), 830-843.

[64] L. X. Wang, *A course in fuzzy systems and control, part 2*, Upper Saddle River, Prentice-Hall International, Inc., 1997.

[65] K. C. Wong, K. S. Leung, M. H. Wong, *Protein structure prediction on a lattice model via multimodal optimization techniques*, In Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, (2010), 155-162.

[66] W. J. Yu, J. Y. Ji, Y. J Gong, Q. Yang, J. Zhang, *A tri-objective differential evolution approach for multimodal optimization*, Information Sciences, **423** (2018), 1-23.

[67] C. Yue, B. Qu, J. Liang, *A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems*, IEEE Transactions on Evolutionary Computation, **22**(5) (2017), 805-817.

[68] Z. H. Zhan, Z. J. Wang, Y. Lin, J. Zhang, *Adaptive radius species based particle swarm optimization for multimodal optimization problems*, In 2016 IEEE Congress on Evolutionary Computation (CEC), (2016), 2043-2048.

[69] Z. Zhang, H. S. Seah, *Real-time tracking of unconstrained full-body motion using niching swarm filtering combined with local optimization*, In CVPR 2011, Workshops, Colorado Springs, CO, USA, (2011), 23-28.